COBB

INSIDE DOS

Tips & techniques for MS-DOS & PC-DOS

December 1991
Vol. 2 No. 12

Take advantage of XCOPY's /M switch when copying a large number of files

If you routinely use the COPY command to copy large numbers of files to floppy disks, you may have encountered this message:

```
Insufficient disk space
```

indicating that the diskette is full. If you then inserted a new diskette and reissued the COPY command, you probably were dismayed to learn that DOS didn't pick up where it left off. Instead, it started all over and copied the same files as it did onto the first diskette.

Fortunately, DOS versions 3.2 and later provide a tool, the XCOPY command's /M switch, that allows you to copy files in succession. In this article, we'll show you how to take advantage of this feature of the XCOPY command. First, however, let's take a quick look at XCOPY, and say a few words about how it can make your copy and backup procedures more efficient.

The XCOPY command

As you might expect, the XCOPY command takes the same basic form as the COPY command:

```
XCOPY source target
```

where *source* is the name of the file you want to copy, and *target* specifies the destination of your copied files. If the source file is not stored in the current drive and directory, you can specify the file's drive letter and full pathname. You can also use wildcard characters to copy several files that have similar names or extensions. When specifying the target argument, you can supply any combination of a drive letter, pathname, and filename.

As you may know, the XCOPY command is especially helpful when you want to copy an entire directory (or several subdirectories) of files. To copy a disk's directory structure as well as its files, simply use the XCOPY command along with its /S switch, like this:

```
xcopy sourceDirectory target /s
```

where *sourceDirectory* is the directory whose files and subdirectories you want to copy, and *target* is the name of the drive or directory to which you want to copy the files. When you issue a command in this form, DOS will automatically copy the directory structure along with the files. Using XCOPY, you'd have to create the directories and subdirectories independently before you copied the files.

The XCOPY command's /M switch

As we mentioned, XCOPY offers another switch, /M, that can make your copy operations quicker and simpler. The /M (for *modify*) switch tells DOS to skip over those files that you haven't modified since you last issued the command.

DOS can tell whether you've modified a file by examining the file's archive bit. If a file's archive bit is turned on, DOS knows that you've modified that file since the last time you issued the XCOPY command and will copy it. If a file's archive bit is turned off, DOS will know that you haven't modified the file, and will not copy that file.

Once you copy a file in this manner, the /M switch turns off that file's archive bit. That makes it possible to reissue the command after you've inserted a new diskette without copying the same files again. In other words, when you fill up your first diskette, simply insert a new

Continued on page 11

IN THIS ISSUE

INSIDE DOS

Inside DOS (ISSN 1049-5320) is published monthly by The Cobb Group.

Prices	Domestic	$59/yr. ($7.00 each)
	Outside the US	$79/yr. ($8.50 each)
Address	The Cobb Group 9420 Bunsen Parkway, Suite 300 Louisville, KY 40220	
Phone	Toll-free	(800) 223-8720
	Local	(502) 491-1900
	FAX	(502) 491-4200
Staff	Editor-in-Chief	Jim Welp
	Consulting Editors	Mark W. Crane
		Tim Landgrave
	Contributing Editor	Van Wolverton
	Editing	Clyde Zellers
		Polly Blakemore
	Production	Debbie Lane
	Design	Karl Feige
	Publications Manager	Elayne Noltemeyer
	Publisher	Douglas Cobb

Address correspondence and special requests to The Editor, *Inside DOS*, at the address above. Address subscriptions, fulfillment questions, and requests for bulk orders to Customer Relations, at the address above.

Postmaster: Send address changes to *Inside DOS*, P.O. Box 35160, Louisville, KY 40232. Second class postage is paid in Louisville, KY.

STATEMENT OF OWNERSHIP, MANAGEMENT, AND CIRCULATION

(Required by 39 U.S.C. 3685). 1A. Title of publication: Inside DOS. 1B. Publication No. 10495320. 2. Date of filing: 9/26/91. 3. Frequency of issue: Monthly. 3A. No. of issues published annually: 12. 3B. Annual subscription price: $59 ($79 foreign). 4. Complete mailing address of known office of publication: The Cobb Group, 9420 Bunsen Pkwy., #300, Lou., KY 40220. 5. Complete mailing address of headquarters of general business offices of the publisher: The Cobb Group, 9420 Bunsen Pkwy., #300, Lou., KY 40220. 6. Full names and complete mailing address of publisher, editor, and managing editor: Publisher, Douglas F. Cobb, The Cobb Group, 9420 Bunsen Pkwy., #300, Lou., KY 40220; Editor, Jim Welp, The Cobb Group, 9420 Bunsen Pkwy., # 300, Lou., KY 40220; Managing Editor, Linda Baughman, The Cobb Group, 9420 Bunsen Pkwy., #300, Lou., KY 40220. 7. Owner: Ziff Communications Co., Ziff Investment Partnership L.P., Philip B. Korsant, the address for all of the foregoing being 1 Park Ave., New York, NY 10016. 8. Known bondholders, mortgagees, and other security holders owning or holding 1 percent or more of total amount of bonds, mortgages or other securities: None. 9. For completion by nonprofit organizations authorized to mail at special rates (DMM Section 423.12 only)—The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes (Check one):__Has not changed during preceding 12 months__Has changed during preceding 12 months. (If changed, publisher must submit explanation of change with this statement.) 10. Extent and nature of circulation—A. Total no. copies (net press run): Average no. copies each issue during preceding 12 months, 91,714; actual no. copies of single issue published nearest to filing date, 114,195. B. Paid and/or requested circulation—1. Sales through dealers and carriers, street vendors and counter sales: Average no. copies each issue during preceding 12 months, 100; actual no. copies of single issue published nearest to filing date, 110. 2. Mail subscription (Paid and/or requested): Average no. copies each issue during preceding 12 months, 73,972; actual no. copies of single issue published nearest to filing date, 85,359. C. Total paid and/or requested circulation: (Sum of 10B1 and 10B2) Average no. copies each issue during preceding 12 months, 74,072; actual no. copies of single issue published nearest to filing date, 85,469. D. Free distribution by mail, carrier or other means. Samples, complimentary, and other free copies: Average no. copies each issue during preceding 12 months, 95; actual no. copies of single issue published nearest to filing date, 171. E. Total distribution (Sum of C and D): Average no. copies each issue during preceding 12 months, 74,167; actual no. copies of single issue published nearest to filing date, 85,640. F. Copies not distributed-1. Office use, left over, unaccounted, spoiled after printing: Average no. copies each issue during preceding 12 months, 17,547; actual no. copies of single issue published nearest to filing date, 28,555. 2. Return from news agents. Average no. copies each issue during preceding 12 months, 0; actual no. copies of single issue published nearest to filing date, 0. G. Total (Sum of E, F1 and 2-should equal net press run shown in A): Average no. copies each issue during preceding 12 months, 91,714; actual no. copies of single issue published nearest to filing date, 114,195. 11. I certify that the statements made by me above are correct and complete. Douglas F. Cobb, Publisher.

Getting around the DEL command's verification prompt

I'm a teacher, and I routinely need to delete all the files on my student's diskettes when they are finished with them. Sometimes I'll delete all the files on hundreds of diskettes at a time. Unfortunately, when I use the command

```
A:\>del *.*
```

to delete the files on that many diskettes, DOS' verification prompt

```
All files in directory will be deleted!
Are you sure (Y/N)?
```

can be a real hassle. I'd like to be able to avoid this prompt—it would be a real timesaver for me. Can you suggest a way?

Karl Lilje
San Luis Obispo, California

There is a way to avoid DOS' verification message: You first create a simple text file that contains only the letter *y*. Then you input that text into a batch file that contains the command

```
del *.*
```

Let's take a look at how you create the files you'll use to get around the verification prompt. But first a word of warning: The DEL command is a potentially dangerous command. When you create a batch file that overrides the warning message, you make it even more dangerous. For this reason, you need to be extremely careful when you run the batch file. In fact, it's probably a good idea to test it in a temporary directory before you actually use it to delete any data files.

Creating Y.TXT

As we mentioned, the first step is creating a text file that contains the letter *y*. We'll call it Y.TXT. To create Y.TXT, type

```
C:\BATCH>copy con y.txt
y[Enter]
[F6][Enter]
```

When you do, your screen will look like this:

```
C:\BATCH>copy con y.txt
y
^Z
                1 file(s) copied
```

Creating DELALL.BAT

Now, create the DELALL.BAT batch file you'll use to delete all the files without a warning message. To create DELALL.BAT, type

```
C:\BATCH>copy con delall.bat
del *.* < c:\batch\y.txt[Enter]
[F6][Enter]
```

When you do, DOS will display

```
C:\BATCH>copy con delall.bat
del *.* < c:\batch\y.txt
^Z
        1 file(s) copied
```

In a nutshell, DELALL.BAT takes advantage of the input redirection symbol (<) to input the letter *y* into the

```
del *.*
```

command. Instead of avoiding the warning message, you trick DOS into supplying the *y*.

Now, anytime you want to delete all the files, you can simply type the name of the batch file rather than the command

```
del *.*
```

like this:

```
A:\>delall
```

Notes

Notice that we stored our Y.TXT text file and our DELALL.BAT batch file in the \BATCH subdirectory. It's always a good idea to store your batch files in a subdirectory separate from your data files. Of course, to be able to run those batch files conveniently, you need to include on the path the subdirectory that contains your batch files.

Also, note that we included the path and file names for the Y.TXT file in the DELALL.BAT batch file. If you store your Y.TXT file in a subdirectory with a different name be sure to substitute that pathname in DELALL.BAT.

Finally, this technique puts the input redirection symbol (<) to work. If you're not familiar with DOS' redirection symbols and you'd like to learn more about them, see the article "Redirecting Command Input and Output Using <, >, and >>" in the June 1991 issue of *Inside DOS*.

Another way to display subdirectory names

In the June 1991 issue of *Inside DOS*, you showed a version of the DIR command that lets you display only subdirectories ["Four Tips for the DIR Command"]. As you demonstrated in that article, you can display everything in a directory that doesn't include an extension by typing the command

```
C:\>dir *.
```

and pressing [Enter]. Typically, all of your files will contain extensions, and all of your subdirectories won't. However, if you include an extension in a subdirectory name, the above command won't include it in the directory listing.

Fortunately, DOS provides a way to list only subdirectories: Append the DIR command to the FIND command with the text string "<DIR>":

```
C:\>dir | find "<DIR>"
```

As you know, the FIND command searches for the text you specify in quotation marks. Because DOS includes the text <DIR> after every subdirectory name, this command will cause DOS to display all subdirectories—even those with extensions. (Because it's case-sensitive you must type the test string <DIR> with all uppercase letters.)

Since I keep a lot of subdirectories on my hard drive, I include the MORE filter to tell DOS to display the list one screen at a time like this:

```
C:\>dir | find "<DIR>" | more
```

Also, because this command is clumsy to type at the prompt, I created a batch file called SUBDIR.BAT that includes the following commands:

```
@echo off
dir | find "<DIR>" | more
```

Now, whenever I want to see a list of subdirectories, I simply type

```
C:\>subdir
```

and DOS lists only the subdirectories.

Claude Sailor
Fort Wayne, Indiana

We thank Mr. Sailor for sharing that tip. By the way, if you've upgraded to DOS 5, it's a little easier to display a listing that contains only subdirectories. DOS 5's DIR command includes a switch, /AD, that allows you to display only subdirectories. To display all the subdirectories in a directory—whether or not the directory name includes an extension—type the command

```
C:\>dir /ad
```

For more on DOS 5's enhanced DIR command, see the articles "Using DOS 5's DIR Command to Sort Files" and "Displaying File Attributes with DIR" in the July and October 1991 issues of *Inside DOS,* respectively.

Creating a special-purpose prompt

Many application programs like Lotus 1-2-3 and WordPerfect let you go to the DOS prompt, execute DOS commands, and return to the application at the same point you left it. Although this is a convenient feature, it does present the following problem: When you temporarily go to DOS, it's easy to forget that your application is still active. If you do forget, and restart the same application or turn off your computer, you lose the chance to save the file that was active when you went to DOS and therefore lose all the work you've done since you last saved that file.

Fortunately, you can write a simple batch file that creates a custom prompt reminding you to go back to your application, then restores the prompt to normal when you exit the application. In this article, we'll show you how to write and use that batch file.

How the "go-to-DOS" feature works

When you select the "go-to-DOS" feature from within an application, you never actually exit from the application (although it looks as if you do). A copy of COMMAND.COM, the DOS command processor, resides within the application. Using this copy of COMMAND.COM, you can execute DOS commands while your application remains active. When you're finished using DOS, you usually type *exit* and press [Enter] to return to the application.

The special-purpose prompt

A batch file like the one shown in Figure A creates a special reminder prompt and then loads your application. You'll see the reminder prompt only when you go to DOS using the application's go-to-DOS feature. When you quit from the application, you'll see your normal DOS prompt once again. The last line of the batch file, which restores the normal prompt, is not executed until you exit completely from the application.

Figure A

```
@echo off
prompt Enter EXIT to return to 123$_$p$g
cd\123
123
prompt $p$g
```

This batch file changes the DOS prompt to remind you to return to your application.

An example

The batch file in Figure A is named LOTUS22.BAT and is designed to work with Lotus 1-2-3. Let's look at the file line by line to see how it works.

The first line of LOTUS22.BAT issues the command

```
@echo off
```

which tells DOS to suppress the display of the batch file commands as it executes them. The second line

```
prompt Enter EXIT to return to 123$_$p$g
```

creates a two-line DOS prompt. The special code $_ (underscore), which follows the text *Enter EXIT to return to 123*, tells DOS to start a new line. The special codes *$p* and *$g* tell DOS to display the current drive and directory followed by a greater than sign. The resulting prompt looks like this:

```
Enter EXIT to return to 123
C:\>
```

The third and fourth lines of LOTUS22.BAT,

```
cd\123
123
```

tell DOS to change to the \123 directory and load 1-2-3. Of course, we could omit the third line if the \123 directory were included in our path.

The last line

```
prompt $p$g
```

restores the normal prompt. Although you can define any prompt you want, ours includes the current drive and directory followed by the greater than sign (>).

Modifying the batch file for other applications

You'll need to write a separate batch file for each application that has a go-to-DOS feature. Fortunately, it's easy to modify our simple batch file to work with other applications.

Of course, regardless of the application, the first line of each batch file stays the same. In the second line, you'll want to reword the text in the custom prompt to whatever is appropriate for your application. Beginning with the third line, you should supply the commands needed to load your application. Of course, you'll need to use one command per line. Finally, you'll need to modify the last line of your batch file if you want to restore a custom DOS prompt that's different from the one we've chosen.

You'll want to give your custom batch file a root name that describes the application it launches. For instance, if

you're creating a batch file for Paradox 3.5, you might want to name the file PDOX.BAT.

Using the batch files

Don't forget to save your new batch files in a directory that's included in your path, so DOS can find them when you're ready to execute them. That way, whenever you want to load an application, just enter the name of the associated batch file at the DOS prompt.

If you normally use a menu program to select and load your applications, the menu will probably work just fine with your batch files. Instead of specifying the usual application startup commands in the menu program, specify the batch file name.

Conclusion

When you go to DOS from within an application, you may lose data if you forget to return to the application. In this article, we've shown you how to build batch files that you can run instead of your normal startup commands. When you do, the batch files customize your prompt and remind you to return to your active application.

ADVANCED DOS 5 TIP

Quick formatting diskettes with DOS 5

In DOS 4 and earlier versions, formatting a previously used diskette requires as much time and effort as formatting a new blank diskette. Each time you issue the FORMAT command under DOS 4.0 or earlier versions, it creates a new root directory and File Allocation Table (FAT) for the diskette you're formatting. It also checks for bad areas on the diskette and marks them so that DOS won't try to store good data on a defective diskette sector.

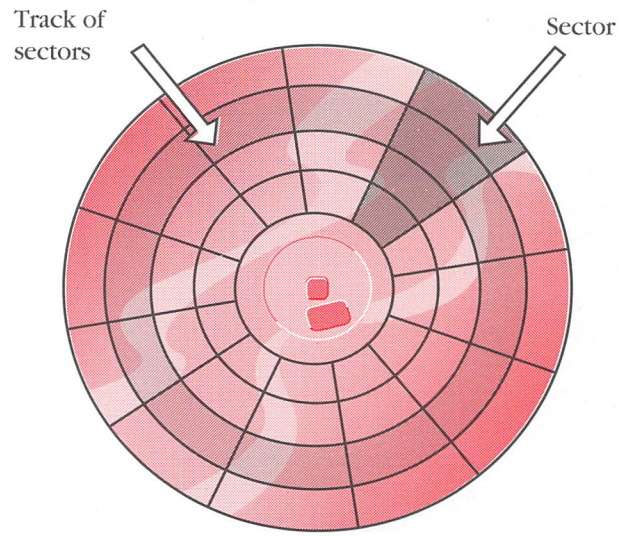
However, the FORMAT command does not erase the data stored on the diskette you're formatting—it just creates a new FAT and root directory for that diskette, thereby making the disk's information irretrievable. (Well, almost irretrievable; DOS 5 provides a new command, MIRROR, that takes a "snapshot" of the FAT and root directory. We'll explain MIRROR in detail in an upcoming issue of *Inside DOS*.) Let's take a look at how DOS stores information on diskettes, then we'll show you the difference between a standard format and a quick format as well as how DOS 5's FORMAT command can speed things up for you.

How a diskette stores information

As you may know, all diskettes store information in concentric circles called *tracks*. Every track is divided into multiple sections called *sectors*. Figure A illustrates this concept. A standard 3.5" double-density diskette has 80 tracks on each side.

In order to keep track of the files stored on each diskette, DOS records the side, track, and sector number where each file begins in a special area of the diskette called the File Allocation Table, or FAT. DOS later uses the information in the diskette's FAT to locate files, much like you use the section, row, and seat numbers on a ticket to find your seat at a ballgame.

Figure A



The FORMAT command divides diskettes into concentric tracks and pie-shaped sectors.

As we mentioned, DOS also creates a root directory when you format a diskette. The root directory stores the name, size, creation date, creation time, and file attributes for the files on the diskette.

Another function DOS performs when you format a diskette is checking for bad sectors on the diskette. A sector can become damaged either physically or magnetically. When DOS finds a bad sector, it stores the location in the File Allocation Table and won't write to that sector of the diskette.

Speeding things up

Fortunately, the DOS 5 FORMAT command offers a new option (/Q for "quick format") that allows you to quickly

reformat a previously formatted diskette. To quick format a diskette in drive A, just type the command:

```
C:\>format a: /q
```

This command tells DOS to create a new FAT and root directory, but not to scan for bad diskette sectors. Since you formatted the diskette and checked for bad sectors the first time, you shouldn't need to check for bad sectors again.

Considering that it still creates a new File Allocation Table and root directory, you might be surprised to learn that DOS can quick format a diskette in about five seconds, compared with the 30 to 45 seconds required for a full-fledged format.

VAN WOLVERTON

Are you still squeaking by without a mouse?

By Van Wolverton

It used to be that you could tell a Mac user from a PC user by the mouse next to his or her computer (not to mention a telltale squint from reading that tiny black-and-white screen). But times have changed: Now Macs have big, color displays, too, and more and more PC users are harnessing a mouse to their machines. Consider it the personal computer version of the harmonic convergence.

From its birth, the Mac dominated the graphics and desktop publishing markets because of its apparently superior display, imaginative software, and, yes, its mouse, which liberated creative users from the clunk-clunk limitations of a keyboard.

But after a few years, graphics programs—and especially computer-aided design (CAD) programs—were released for the PC and these relied on the mouse. In short order, PCs came to dominate the CAD market; only a tiny percentage of PC owners used a CAD program, however, so the number of systems with a mouse was still small. The early versions of Windows tempted a few more PC owners into buying a mouse, but most languished in disuse for lack of PC software that took advantage of the mouse.

But with the release and success of programs such as Aldus PageMaker, Ventura Publisher, PC Paintbrush, Corel Draw, and dozens of others, PC owners could finally experience for themselves the appeal of mousing around. No longer was a mouse considered a curiosity beside a PC. When Microsoft released Windows 3 in 1990, the die was cast: It was to be a mouse world, after all.

Apple didn't invent it

Apple didn't invent the mouse (nor much else of the look and feel usually called "Mac-like"). The mouse was first developed in the mid-70s at Stanford Research Institute (now called Stanford Research International), a think tank loosely affiliated with Stanford University. Researchers there carved the first prototypes from wood and wired them with up to five switches. In a feat of manual dexterity equaled only by gifted violinists and Nintendo champs, some adepts at SRI would actually enter text one character at a time by pressing the mouse key combinations corresponding to the ASCII code of the letters and numbers.

This obviously wasn't a technique destined for the mass market, but the basic notion of a mouse was adopted by Xerox at its Palo Alto Research Center and incorporated into a desktop computer system called the Xerox Star. Released in 1981, the Star used a mouse, drop-down menus, symbols to represent files and functions (Xerox dubbed them *icons*), and a white screen with black text and graphics. The Star was not a successful product, but then Apple, in turn, adopted the technology for its nascent Macintosh computer, and the rest is, well, ironic.

Do you need a mouse?

Many early users of the mouse preferred it to the keyboard because they weren't skilled typists. But keyboard avoidance is the least of the reasons for using a mouse. A program designed to exploit the capabilities of the mouse lets you use the mouse for virtually everything except entering data. In particular, you don't have to memorize commands, type file names, or otherwise behave as if you were preparing for a midterm exam.

Graphics programs virtually require the use of a mouse. It simply isn't practical to use the cursor-control keys to operate a program that involves freehand drawing or manipulation of graphic elements, such as for draw, paint, and CAD programs. Not only is a mouse faster, but it also lets you apply the right side of your brain—the intuitive side—to the creative projects for which these programs are often used. You're free to think about what you're creating, not about how to use the program.

The main disadvantage of using a mouse is the need to switch back and forth between the keyboard and mouse. Letting go of the mouse and moving your hand back to the keyboard (and the reverse) is awkward for most of us; for a dedicated touch-typist, it's downright irritating. Most graphics programs minimize the need to switch; it's possible, in fact, to get lost in your work for hours without ever touching the keyboard.

Not always the best

The mouse isn't a panacea. It offers far fewer advantages when you're using a word processor, unless it's one of the more recent ones designed for use with a mouse, such as Microsoft Word for Windows or Ami Pro. Even with those, the mouse offers no advantage when you're entering text. Many people do use the mouse to move around in a document and select passages when they're revising or formatting text, but fast typists argue that switching from the mouse to keyboard and back while revising and formatting text is slower than using the keyboard for everything.

Graphics-oriented spreadsheets such as Microsoft Excel, Quattro Pro, and Lotus 1-2-3 for Windows fall into the same category as word processors: The mouse isn't much help when you're entering data or formulas, but you can make good use of the mouse when you're formatting the spreadsheet and working with the graphing part of the programs.

The often-mentioned problem of finding the space to roll a mouse around on a cluttered desk is highly overstated. Windows—and most other programs designed to use a mouse—let you control the sensitivity of the mouse so that you need only a few square inches for full-screen movement.

Serial vs. bus mouse

There are two mouse variants: *serial* and *bus*. A serial mouse plugs into a serial port, occupying a valuable connection that you might need for a modem or serial printer. A bus mouse, on the other hand, takes up an equally valuable expansion slot inside your computer, which you might need for memory expansion, a disk controller, or a fax modem. (IBM PS/2 machines and a few others have a special mouse connector that accepts a bus mouse without taking up either a serial port or an expansion slot.)

The serial mouse is easier to attach because you don't have to remove the cover of the machine. To install a bus mouse, however, you must take off the cover, insert the card, and replace the cover; in rare cases, you might have to change the settings of some switches or jumpers on the mouse card to eliminate interference with another device.

You may not have to choose which mouse type to buy. Some stores and mail-order companies include a mouse with their systems, so the choice is made for you. Depending on what accessories you have added to your system and how many serial ports and expansion slots it includes, you may have room left for only one type of mouse. If you have room for either type, there doesn't seem to be much difference in speed or resolution. Unless you have only one available serial port and plan to use it for something else, the serial mouse is probably a better choice simply because it's easier to install.

Should you get a mouse?

If you have Microsoft Windows or are planning to try it, you should definitely get a mouse. There's no way to use Windows itself to its fullest potential—nor most of the programs designed specifically for Windows—using just the keyboard. The DOS 5 Shell—an eminently useful alternative to the DOS command line—also lets you use the mouse for most routine program- and file-management chores. If you plan to use a graphics program or try your hand at desktop publishing, you'll need a mouse to realize the potential of your programs. CAD programs, too, are almost unusable without a mouse.

Most other programs don't offer such a clear-cut choice, primarily because with word processors, spreadsheets, and database programs you spend a significant amount of time entering data at the keyboard. There's no way to avoid switching back and forth between the keyboard and mouse, but you might find that the mouse makes certain functions of the program much easier to use.

Whatever programs you use now, the trend in software design is increasingly toward the use of pop-up menus, symbols, and other graphic techniques you can exploit to the fullest only with a mouse. Although virtually all these programs offer—or will offer—a keyboard alternative for every function, the mouse is clearly superior in most cases. If you haven't yet tried a mouse, it's probably time to take the plunge.

Contributing editor Van Wolverton is the author of the best-selling books Running MS-DOS 5 *and* Supercharging MS-DOS. *Van, who has worked for IBM and Intel, currently lives in Alberton, Montana.*

DOS 5 SHELL TIP

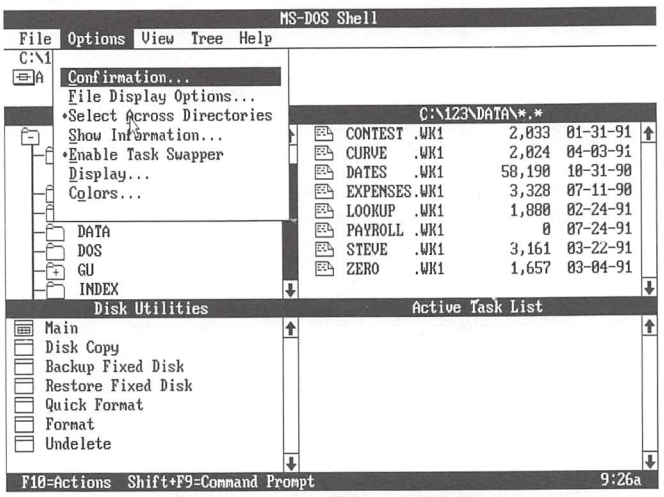
Use caution with the Select Across Directories command

DOS 5's Shell offers some file-manipulation advantages over the system prompt. For instance, by clicking and dragging with the mouse, you can easily move files among directories or disks. Similarly, you can copy, print, delete, and rename files easily in the Shell.

Before you can work with files, however, you must first highlight or *select* the files in the File List window. To

this end, DOS provides many selection tools to make working with files easier. (For more information on selecting files, see the article "Managing Files with DOS 5's Shell" in last month's issue of *Inside DOS*.)

Figure A



The Select Across Directories command allows you to select files in different directories before you issue a command.

One command, though, can get you into trouble: the Select Across Directories command on the Options menu, shown in Figure A. When you choose Select Across Directories, you can select a file or group of files, then activate a second directory and select files there. When you've selected all the files you want to work with, you issue a menu command just as you normally do. DOS will then carry out that command on all the files you've selected. Obviously, this can be a handy feature when you want to manipulate a large number of files.

Unfortunately, it's easy to lose your place when you select across two or more directories. Because DOS can only show the contents of one directory onscreen at a time, you might forget which files you've selected in the original directory. Or, you might inadvertently select files, then move to another directory unaware of the original selection. As you might imagine, this can lead to disaster if you delete files. Likewise, if you accidentally move program files out of their directory, you might not be able to find and replace them.

Although the Select Across Directories command is turned off by default, it is a toggle command. That means once you activate it, it stays on until you turn it off again.

For these reasons, it's a good idea to exercise caution whenever you use the Select Across Directories command. Also, any time you use the command to select files in different directories, we recommend that you turn it off again when you're finished.

BATCH FILE TIP

Displaying a blank line in a batch file

When you write a batch file that displays text on the screen, you'll usually want to insert a blank line to make the text more attractive and easier to read. For instance, if you create a menu system for your computer, you'll want to insert a blank line between the menu headings and the options on each menu. In this article, we'll show you how to use the ECHO command to create a blank line in your batch files.

The problem

As you probably know, the ECHO command displays text on the screen. For example, the command

```
echo Good Morning!
```

displays the line

```
Good Morning!
```

As you may have discovered, however, you can't echo a blank line to the screen by simply typing the command

```
echo
```

If you do this, DOS displays either the message *ECHO is on* or the message *ECHO is off* instead of displaying a blank line.

The solution

Fortunately, you can use the ECHO command to display a blank line by echoing one of DOS' nonprinting characters to the screen. All you have to do is immediately follow the word *echo* with the appropriate character. For versions 3 and later, all of the following commands will produce a blank line:

```
echo.
echo+
echo/
echo[
echo]
echo:
```

For earlier DOS versions, you'll need to follow the ECHO command with one of the following keys or characters:

tab
comma

line feed
space
semicolon
equal sign

as well as a trailing space. For instance, under DOS 2.1 you can produce a blank line by typing *echo*, immediately followed by a tab and a space. (Notice that later versions of DOS don't care whether you follow the nonprinting character with a space, while earlier versions require the trailing space.)

If you want to write a batch file that generates a blank line in nearly every version of DOS, try typing the word *echo*, pressing the [F7] key, and then typing a space. Pressing the [F7] key produces an @ sign in your batch file, but DOS interprets this character as a nonprinting character and therefore generates a blank line.

An example

Figure A shows a sample batch file that displays two lines of text separated by a blank line.When you run the batch file in Figure A, DOS will display the text

```
Data Entry Menu

Enter your selection
```

Figure A

```
@echo off
cls
echo Data Entry Menu
echo]
echo Enter your selection
```

This batch file produces a heading, a blank line, and a line of text.

Printing multiple blank lines

You might think you could produce several blank lines in succession by following the word *echo* with several nonprinting characters. Unfortunately, this technique doesn't work. For instance, if you include in a batch file the command

```
echo]]]]
```

and then run that batch file, DOS will display the characters

```
]]]
```

instead of a blank line. For each blank line, then, you need to use a separate ECHO command that echoes a single nonprinting character.

DOS 5 TIP

Using SETVER to run your applications with DOS 5

We'd like to thank Microsoft's Product Support Services staff for providing us with much of the information for this article.

If you've upgraded to DOS 5 and tried to load your applications as you normally do, chances are you've encountered the

```
Incorrect DOS version
```

error message. You might see a message like that because many application programs are designed to run only under a particular version of DOS. When you try to start such an application, it will typically check with DOS to see which version of DOS is running before it loads and runs. If DOS reports a version number other than that for which the application was designed, the application will not run.

For instance, if you attempt to start an application designed to run under DOS 3.3, and DOS reports to that application that the current version is 5, the application will not start. Instead it will display a message such as *Incorrect DOS version*.

If you see a message like this when you try to start an application under version 5, two possibilities exist:

1. The application isn't compatible with DOS 5.

2. The application *is* compatible with DOS 5, but won't run unless it interprets version 5 as the version it was designed for.

If you find yourself in this situation, you should contact your application vendor to find out whether the application is compatible with DOS 5. If not, you need to acquire an updated version of the application from your vendor.

However, if the application *is* compatible with DOS 5, you can use a DOS command, SETVER, to "trick" the application into thinking that it's running under a previous

version of DOS. To do this, you simply add the application's name to DOS 5's *version table*, which lists the applications for which DOS must report a version number other than 5.

Loading SETVER.EXE

DOS stores and maintains the version table in the file SETVER.EXE. To load SETVER.EXE, you need to include the command

```
device=c:\dos\setver.exe
```

in your CONFIG.SYS file. As you may know, DOS 5's Setup program automatically adds this command for you when you install DOS 5 on your system. As long as this command appears in CONFIG.SYS, DOS installs SETVER.EXE and loads the entire version table each time you restart your computer.

Adding an application

To add an application's name and its associated version number to the version table, issue a command that takes the form

```
setver filename n.nn
```

where *filename* is the name of the application file (including its extension), and *n.nn* is the version number you want DOS to report to that application. (By the way, you don't need to include the drive or path in the *filename* parameter.) Once you've issued the SETVER command, restart your computer by pressing [Ctrl][Alt][Del] to put your change into effect.

For example, suppose you use an application named EASYED.EXE, which requires DOS to report version 3.31. To run EASYED.EXE under DOS 5, you add its name and associated version number to the version table using the command

```
C:\>setver easyed.exe 3.31
```

Then, you restart your computer by pressing [Ctrl][Alt][Del].

Fortunately, SETVER lets you display the current contents of the version table. To do this, issue the command

```
C:\>setver
```

When you display the contents of the version table, you'll notice that DOS 5's Setup program has added to the version table several programs and device drivers.

Removing an application

Whenever you install a new version of an application that is compatible with DOS 5, you'll want to remove that application's name from the version table. To do this, issue a command that takes the form

```
setver filename /d
```

where *filename* is the name of the application file you want to remove and the /D switch tells DOS to delete the application's name from the version table. As before, you'll need to restart your computer in order to put your change into effect.

Keep in mind that the SETVER command does not affect whether an application is compatible with DOS 5. Instead, SETVER works only for applications already compatible with version 5, but must see a different version number when they ask DOS to report which version is running.

When you change the version table with SETVER, DOS displays a warning message reminding you first to verify that the program you're attempting to run is compatible with DOS 5. If you don't make sure your program works with DOS 5, you could lose data.

DOS 5 SHELL TIP

Renaming directories with the DOS 5 Shell

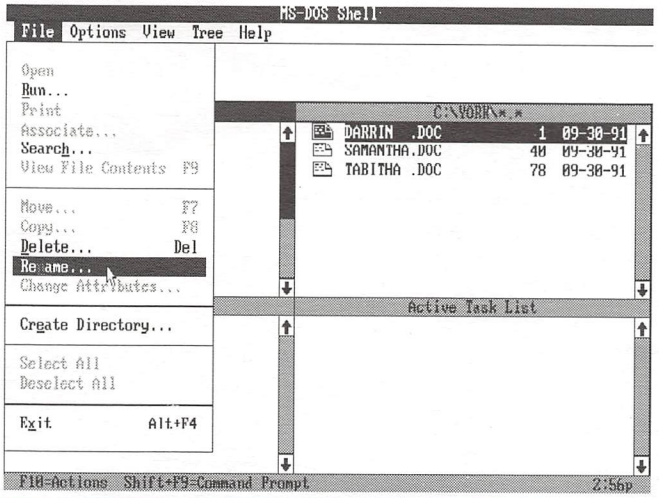
DOS 5's Shell, with its graphical interface, offers a new way of working with DOS. As we've shown in previous issues, the Shell simplifies some tasks by allowing you to click and drag with the mouse. Also, the Shell's pull down menus keep commands at your fingertips so you don't have to memorize command names and syntax. These features are especially helpful to DOS newcomers.

However, the Shell isn't just for newcomers or mouse users. Perhaps as an incentive to encourage users to become familiar with the Shell, Microsoft included a few commands that aren't available at the prompt. Using these commands, you can accomplish what would otherwise require a series of prompt commands.

The Rename... command

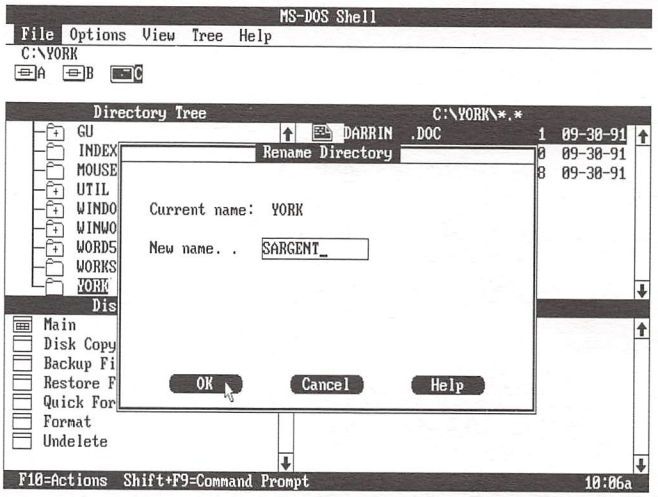
One such command is the File menu's Rename... command. As you know, the system prompt provides a command, RENAME (or REN) that allows you to rename files. However, the RENAME command doesn't allow you to rename directories or subdirectories. If you create a lot of temporary subdirectories, or need to rename subdirectories often, you know what a hassle it can be. To rename a subdirectory at the prompt, you must first create

Figure A



Unlike the prompt's RENAME command, the Shell's Rename... command allows you to rename subdirectories.

Figure B



Just type the new name in the Rename Directory dialog box and press [Enter], and DOS will rename the subdirectory.

a subdirectory with the new name you want to use. Next, you must copy all the files from the old subdirectory to the new one. Then, you must delete all the files in the old subdirectory. Finally, you must delete the old subdirectory with the RD (Remove Directory) command.

However, the Shell's Rename... command allows you to rename a subdirectory in a flash. Before you can take advantage of this command, load the Shell by typing

```
C:\>dosshell
```

(This command assumes you've included in your path the directory that contains your DOS system files.)

To rename a subdirectory, first select the subdirectory's name in the Directory Tree window by clicking on it with your mouse or pressing [Tab] to activate the window and ↓ to select the subdirectory. Next, pull down the File menu and issue the Rename... command with your mouse, or press [Alt]F and then press N. When you do, DOS will present the Rename Directory dialog box.

The dialog box shows you the directory's current name and provides a text box for you to enter the new name. Simply type in the new name and press [Enter] or click OK. Immediately, DOS will rename the subdirectory.

For example, suppose you decide to change the name of a subdirectory from YORK to SARGENT. Begin by selecting the YORK subdirectory. Next, pull down the File menu and issue the Rename... command as shown in Figure A. When DOS presents the Rename Directory dialog box, type SARGENT, as shown in Figure B. Finally, press [Enter] or click OK to lock in the change. When you do, DOS closes the dialog box and renames the subdirectory. That's all there is to it!

Conclusion

The DOS 5 Shell offers a few features that aren't available at the prompt. Among them is the ability to rename subdirectories with a single command. In this article, we examined the Rename... command and showed you how to rename a subdirectory.

Continued from page 1

Take advantage of XCOPY's /M switch...

diskette and issue the XCOPY command again. When you do, DOS will skip over all the files you copied onto the first diskette because the initial XCOPY command turned off the archive bits. (By the way, the XCOPY command requires blank formatted diskettes.)

Using XCOPY instead of BACKUP

If you use DOS' BACKUP command in your backup routine, you know that this command compresses your files, enabling you to store more information on each floppy diskette. Unfortunately, you must use the RESTORE command to access those files. For this reason, many users prefer XCOPY to BACKUP because they don't want the hassle of using the RESTORE command to uncompress files. Because the /M switch can tell XCOPY which files haven't been modified since the last time you backed up, it's a good alternative to BACKUP if you need to back up only a few files or if you often need to access your backup files.

Please include account number from label with any correspondence.

For example, to back up to drive A all the files in all the subdirectories on your hard drive that you've modified since your last backup, issue the command

```
C:\>xcopy c:\*.* a: /s /m
```

When DOS fills up the first diskette and returns you to the prompt, press [F3] (or ↑ if you're running DOS 5 and have Doskey loaded) and [Enter] to reissue the command. DOS will continue where it left off because you included the /M switch in the XCOPY command.

Copying every file

As we mentioned, the XCOPY command's /M switch only copies the files whose archive bits are turned on. However, sometimes you'll want to copy every file, including those you haven't modified. You can accomplish this by using the ATTRIB command to turn on every file's archive bit. Before you issue the XCOPY command, issue the ATTRIB command in the form

```
C:\>attrib +a c:\*.* /s
```

The ATTRIB command's +A parameter tells DOS to turn on the archive bit; the /S switch tells DOS to look in each subdirectory.

Once you've issued this command, you can begin the process of copying every file on the hard disk using the XCOPY command. When DOS fills up the first diskette with files, it will present the *Insufficient disk space* message we mentioned earlier and return you to the system prompt. At that point, you can replace the diskette in drive A with a new one, and press [F3] (again, you can press ↑ if you're running DOS 5 and have Doskey loaded) to recall the XCOPY command. When you press [Enter], DOS will pick up where it left off, and continue copying additional files from drive C to drive A.

By the way, if you want to exclude only a file or two from your universal copy procedure, you can do so by turning off that file's archive attribute with a command in the form

```
attrib -a filename
```

where *filename* is the full pathname of the file you want to exclude from the backup. For example, suppose you want to make copies of all the files on drive C, except for the file C:\123\PAYROLL.WK1. To do this, you would use the following three commands:

```
C:\>attrib +a c:\*.* /s
C:\>attrib -a c:\123\payroll.wk1
C:\>xcopy c:\*.* a: /s /m
```

The first command turns on the archive bit of every file on drive C. The second command turns off the archive bit of the file C:\123\PAYROLL.WK1. Finally, the third command tells DOS to copy to drive A all the files on drive C whose archive bits are turned on.

Conclusion

DOS 3.2 and later versions include the XCOPY command. XCOPY provides a couple of useful switches, /M and /S, which you can use to overcome some of the limitations of the COPY and BACKUP commands. In this article, we examined XCOPY and its /M and /S switches.

XCOPY's other options

In the article "Take Advantage of XCOPY's /M Switch When Copying a Large Number of Files" on page 1 of this issue, we demonstrated the XCOPY command's /M and /S switches. Here's a quick look at XCOPY's other switches:

/A	copies only files that have their archive bits turned on. This switch works just like the /M switch except it leaves the archive bits turned on when it copies the file.
/D:*date*	copies only the files modified on or after the date you specify in *date*.
/P	prompts you to confirm the copy operation before it copies each file.
/E	copies empty subdirectories.
/V	verifies each file as it copies it. The /V switch checks to make sure each copy is identical to the original file.
/W	prompts you to press any key before it begins the copy operation.